A2-Central™

formerly Open-Apple

A journal and exchange of Apple II discoveries

September 1992
Vol.8, No.8

ISSN 0885-4017
newstand price: $3.00
photocopy charge per page: $0.20

How to be an Apple II Programmer

by Matt Deatherage

In an age where "user friendliness" is prized above all, when software companies are doing backflips trying to figure out how to give users power without requiring extensive technical knowledge, and when the complexity of machines like the Apple IIgs are spawning stacks of technical manuals bigger than 98% of all household pets, why on earth would anyone want to be a programmer?

Those of you who like to program already know why—it's like asking "why bother to learn to play the piano when compact disc players are so inexpensive?" Using other people's work just can't compare to creating your own. Guiding an idea from your mind to your screen to the screens of thousands of people around the world is a lot of fun, and if your idea is particularly useful, you can make some decent pocket change on the side. Some people who've had incredibly useful ideas even manage to make a living at it.

Or sometimes you just want a little, teeny program to do one specific thing on your computer. Let's say you have about 400 floppy disks with essays, notes and other interesting files on them, and you want a way to catalog them easily. You might try writing a little Applesoft BASIC program to put the catalog of each disk into a text file, so you can read it into AppleWorks and print it out. So you do that, and it doesn't take all that long, and you like it.

But it could be better.

So you spend some more time fixing it to only catalog AppleWorks files. Then you realize that the file names don't tell you enough, so you put in a little code to read the first few lines of any text file or AppleWorks file and put that in your text file as well. That's very helpful, but then you need to reformat the output so it's easier to read. So you do that—but then you realize you want to search the catalog file. So your program starts writing a second text file that contains an index of the text in the first file, so you can find files based on the first few words. But that index isn't too useful (every file contains the word "the", for example), so you start adding a way to input keywords to better identify files.

Pretty soon you look up from your screen, you find that three weeks have passed, you've lost 30 pounds and now your program catalogs every file on every disk by name, type, keyword and modification date, creates a compressed AppleWorks Data Base file pre-sorted by age and operates so quickly you can't feed it disks quickly enough to catalog.

This leads to the third reason you might want to be an Apple II programmer—you could be brain-damaged and not know any better. This is how Nifty List was created.

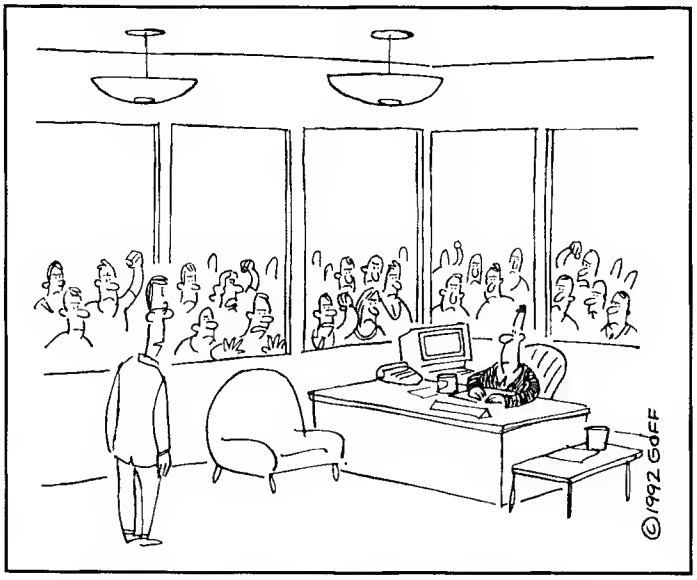
In this article, I attempt to explain some of the mysteries about how to be a programmer. Note that I'm not telling you how to program—one article on "how to program" is like one article on "how to paint"—it can tell you how to hold a brush but not how to create art. Instead, I'm going to discuss all the aspects of being a programmer except the actual programming. Not getting the these details right has foiled more than one aspiring software artist, who thought that knowing how to program was all any programmer needed. It's not.

Why do programmers need anything? Lots of programmers may think they don't need anything. They think that they can sit at the keyboard and overcome any software or hardware obstacles with a superior intellect, or by sheer willpower, or, if necessary, with a sledgehammer. And maybe 15 years ago this was true—but not today.

The people who might want to use your programs have a staggeringly vast selection of hardware and software on their systems. Hard drives, networks, accelerators, alternative input devices, cards to assist the physically challenged and more. And on the Apple IIgs, your programs have to be compatible with whatever initialization files, desk accessories or control panels the user might have. A person sitting alone in a room with no assistance doesn't stand a chance against all this.

Fortunately there are lots of things in the world to assist you, ranging from the obvious (like programming manuals) to the not-so-obvious (like a modem). All these tools help you with your responsibility as a programmer.

When you write a program for your computer, you're playing with your system. It's yours. You bought and paid for it, so have a ball. Splurge. Do whatever you feel like (but make sure you have backups first). However, if you give a program you wrote to someone else, you're playing with their system. That's different. Apple and a lot of third parties have worked very hard to provide guidelines for program-



"SOME USERS ARE HERE TO DISCUSS YOUR DECISION NOT TO UPGRADE OUR PRODUCT."

mers to follow so that what works on your system can work on everyone's system. As a programmer, it's your responsibility to follow these guidelines so your program doesn't do any damage to someone else's system.

If you give the program to a few friends, you can know how their systems work and warn them of problems ahead of time. But if you distribute your program widely, you don't get that luxury. Fortunately, following compatibility guidelines makes you as certain as you can be that your program will work for everyone, not just for you and your friends.

Every computer user has, at one time or another, found a program they really, really wanted to use on their system just to find out they couldn't make it work, no matter what they did. One of my early experiences was with the game *Alien Mind*, which would not run on my system no matter what. I eventually had to return it for a refund. It turned out that the program used copy-protection schemes specific to the Apple 3.5 Drive, and I don't own one. I own UniDisk 3.5 drives, and I couldn't run the program.

Remember such a time in your experience, and remember how frustrated and angry it made you—and remember it while you program. Your program could be the one someone else wants to use more than anything else, and if you follow the guidelines they'll get to use it. Don't deprive other users the chance to experience your genius.

The first thing you need is a development system. That's pretty easy—that's the system you program on. The more programming you do, the smarter you are to make it as powerful as possible. If you don't have an Apple IIgs, you ought to get one. Just out of the box, it's two and a half times faster than any other Apple II (except an Apple IIc Plus), plus it's expandable, has 1 MB of memory built-in and extra features. A great number of your users will have Apple IIgs computers as well, so you can test more easily that way. If you have a reasonable fast Macintosh, you might consider using Apple's cross-development tools, letting your Macintosh do all the work of translating your program into Apple II executable code while your Apple II computers test. The cross-development tools are generally very fast, so they can add to your productivity. Having two machines speeds things up a lot, no matter what the second machine is—but if it's another Apple IIgs or a Macintosh, you can compile for one machine while the other reboots. That's awfully handy if you have the resources for it—but if you don't, don't worry about it. One machine will do for 99% of what you want to accomplish.

If you don't have a hard drive, get one. Some things that make development a lot easier (like having really big development environments and access to a lot of files at once) just aren't practical using only floppy disks. If you find yourself twiddling your thumbs while your program recompiles, an accelerator would do you a great deal of good. And if you have multiple machines, an AppleTalk network is great for testing and for moving things back and forth.

Isn't this all a lot of money? Not really, considering what you get. If an accelerator costs you $200 but makes your machine compile three times faster than it used to, how long would it take before all that extra speed makes you more productive? How much is $200 of your time? A few hours? A few days? An accelerator, or a hard drive, or maybe a network, will save that much time within a few weeks and continue to do so forever. If you just want a network for testing, on an Apple IIgs you can make one really cheaply by installing AppleTalk and connecting to a friend's computer with a regular printer cable, or to a Macintosh, or to nothing. You still have full access to AppleTalk services even if you're the only computer on the network, like you are when no network is attached. Handy for testing.

The more programming you do, the more a high-powered development system will pay off. You don't have to have any of these things to program, especially if you don't program a lot, but they'll help in other ways. Don't feel pressured—get the ones that will help you the most as you're able to get them. Anything that makes your development system more powerful helps you in the long run.

Next you need to get a development environment if you don't have one. What you choose depends on how you want to program.

If you program the Apple IIgs in Pascal or C, you almost certainly want *ORCA* products from the The Byte Works, Inc. (4700 Irving Blvd NW Suite 207, Albuquerque, N.M. 87114 505-898-8183). *ORCA/Pascal* and *ORCA/C* are the leading high-level language compilers for the Apple IIgs, and they integrate well with *ORCA/M*, the Apple IIgs assembler standard. Each comes with a full development shell and everything you need to start writing programs.

If you write in assembly, *ORCA/M* is a good solid choice. Many people swear by *Merlin 16+*, an assembler published by Roger Wagner Publishing (1050 Pioneer Way Suite P, El Cajon, Calif. 92020, 619-442-0522). *Merlin 16+* is considerably faster than ORCA/M, but it's more cryptic and takes a longer time to learn. It also stores your source code in non-standard text files, and it's harder to use *Merlin* routines with high-level languages like *Pascal* or *C*. If assembly's all you want to use, *Merlin's* worth a second look just for the speed.

If you write software for 8-bit Apple II computers, there have been hundreds of tools developed over the past 15 years to assist. Not only can most of the Apple IIgs assemblers *(ORCA/M, Merlin, MPW IIgs)* compile 6502 code optionally, but there are lots of other tools available as well, for debugging and managing Applesoft and other source code and more. The Resource Central catalog is the place to look for such tools; if you don't have a current catalog, call Resource Central (913-469-6502) and ask for one. They're very handy.

The second most important thing every programmer needs is information. Information is the lifeblood of the computer industry; most of what people do with computers is manage information. You, as a programmer, need to know what people want, and when they want it, and how to do it. If you give all your friends a great program that prints their Christmas Card mailing labels and they all love it, but they all want it to read AppleWorks Data Base files, you can't do what they want unless you know how to read AppleWorks Data Base files—and surprisingly few people are born with this capability. If you can't find out how to do it, you can't fill your friends' needs (and hopefully relieve their overburdened wallets of a little Christmas spending cash).

The official compatibility information usually comes from Apple Computer, Inc., and it comes in several formats. The most well-known are the technical manuals, many of which are published by Addison-Wesley and some of which are available only from Resource Central.

There are lots of great third-party books available about programming the Apple II, and I recommend you read any of them that interest you. But only Apple's manuals and the ones Apple blesses contain official compatibility information, and it's important to know the difference. Apple works pretty hard to make sure that the official compatibility information is true and stays true—Apple can't guarantee what other people say. For example, suppose you have a book that says the Apple IIgs system software routine GetNextEvent is located at a certain address in ROM. That's fine, and it's probably true, or at least it was true when the author wrote the book. However, if Apple's official manual (in this case, Volume 1 of Apple IIgs Toolbox Reference) doesn't say you can do it, then you can't. If Apple doesn't say it works, Apple's not guaranteeing it will continue to work in the future, or that it will work for everyone. That's the difference, and it's important to remember.

The Apple IIgs manuals are the most prolific. If you want to use any toolbox routines, you need the three volumes of *Apple IIgs Toolbox Reference*. If you want to use GS/OS calls, you need *GS/OS Reference*. Those who use the firmware or hardware directly will want *Apple IIgs Firmware Reference* or *Apple IIgs Hardware Reference, Second Edition*. Apple's official reference for ProDOS 8 is the *Pro*

DOS 8 Technical Reference Manual, and BASIC.System's reference is *BASIC Programming with ProDOS*. For using Apple's floating-point environment, you'll need *Apple Numerics Manual, Second Edition*. Anyone who wants to use features in Apple IIgs system software later than version 5.0.2 (that includes the new 6.0) needs *Programmer's Reference for System 6.0*, which is an officially-Apple blessed manual even though it's published by The Byte Works. There are other manuals available, but those are the most common—and most programmers won't need all of those books, at least at once.

Apple updates the manuals and provides new information through a series of documents called *Apple II Technical Notes*. Technical Notes are official word from Apple as well and every programmer who wants to do the right thing will always have a complete set of them. They're available for free from many user groups or online services, although the online services usually only have text file versions that don't read nearly as clearly as the printed, typographical paper versions. Those of you who subscribe to ***A2-Central On Disk*** probably prefer reading the articles in the newsletter instead of on your screen (or even printed on your printer)—the same is true for Technical Notes. You can get paper versions from most user groups, or from Resource Central.

Technical Notes have a sister product called *Apple II File Type Notes*, where Apple details the information on file formats for those file types and auxiliary types they have permission to publish. File Type Notes also contain the current public list of assigned file types and auxiliary types, as well as information on getting assignments from Apple for your own programs. Anyone can get an assignment from Apple, and you don't have to publish your file format to do so. Details are in About File Type Notes, and File Type Notes are available wherever Technical Notes are.

It might seem like all these manuals are expensive, and they're not free—but fortunately, they don't come out all that often. *Programmer's Reference for System 6.0* is the first new required manual since *GS/OS Reference* was released in August 1990—that's not very frequent.

One thing you must be careful about is to always use current information. Apple and others may make "beta", or pre-release, versions of technical documentation available before the final books are ready so you can get a head start programming new features. If you want to use beta documentation, you can—but as soon as the final documentation comes out, you must discard the beta stuff and get the finished product. The finished manuals always contain lots of changes over the beta product, and any one of these subtle changes could make your software misbehave. You may have to pay for a finished manual again even though you bought the beta version not too long ago—that's life in the computer lane. If you don't want to pay for it twice, just don't buy the beta documentation. Using old reference material doesn't save you any money in the end—the time you'll spend tracking down one bug fixed in the final book would have easily paid for that book.

The other kind of information you'll regularly need is information about your development system. You rely on those products (your hard drive, accelerator, assembler and compiler) to produce your software. If there's a newer, better version available, you want to use it. Better yet, if there are bugs fixed, you need to know about them. You also want to know about new hardware products that can aid your development—faster and larger hard drives, or hardware debuggers, for example.

Using current development tools is as important as using current manuals. Update fees are usually very nominal—Byte Works charges $10 for an update to a language. You can spend the $10 or you can spend hours trying to figure out why your program won't work when you might have encountered a bug fixed in the latest version—which is a better use of your time and your $10? Personally, I order the updates.

Information is vitally important, and here's how you get it. Most of the information is pretty easy to come by—you just need to pay attention to a few sources and it will flow to you like a river. There are three major sources of development information—magazines, online services and user groups.

A2-Central is a great way to keep up with what happens in the Apple II world; you'll rarely find eight pages of more important information each month. And reading ***A2-Central*** can keep you up-to-date on lots of products and trends in the Apple II world, as well as letting you know when new technical material is available, but it's not a programming journal.

Anyone programming for the Apple IIgs is cheating themselves if they don't subscribe to *GS+ Magazine (EGO Systems P.O. Box 15366 Chattanooga, TN 37415-0366)* . *GS+* not only contains reviews and news of the Apple IIgs community, it also has programs written by the *GS+* staff and articles on how they work. Those who also get the bimonthly disk get the source code as well, which is hard to beat.

Although it's not a programming journal, *A+/inCider* is still the largest Apple II magazine out there. It won't tell you how to debug your program, but reading it carefully will tell you a lot. *A+/inCider* is geared towards less technical users, so what you read in their pages is what a lot of people are having trouble understanding and need explanations about. You can also notice Apple II market trends in their pages, and these things are wise to heed.

However, without question, the fastest, most up-to-date source of development information are the online services. Any programmer who wants to be a good programmer will realize, soon after joining any active online service, that a modem is not an optional item for a programmer. You can make a local phone call in most parts of the country and get nationwide access to the very latest information, news and development issues that affect you. You can talk to customers, and if you want to distribute your programs as freeware or shareware you can distribute them from your own computer room. It is, without question, the most important way to stay in touch.

The least expensive services to peruse are your local bulletin board systems. You can often find late-breaking news and information on there, but there are some drawbacks. Most BBSs aren't programmer-oriented, while each of the major online services has an Apple II programmer-specific area. Much of the news you'll find locally is second-hand, while the national online services have the direct scoop. But local BBSs are usually free, while national services charge about $6/hour for your online time.

It's worth it, no matter which service you choose. You can ask programming questions and find people from Apple Computer answering them. You can ask a question about the *ORCA* languages and you might find Mike Westerfield (the author) responding. You can download the latest sample code, Technical Notes and news any time, day or night—and you only choose what you want.

There are three main online services for programmers—GEnie (401 North Washington Street, Rockville, MD 20850, 800-638-9636), America Online (8619 Westwood Center Drive, Vienna, VA 22082, 800-827-6364) and CompuServe (P.O. Box 20212, Columbus, Ohio 43220, 800-848-8199). There are other services as well (Delphi, 1030 Massachusetts Avenue, Cambridge, Mass. 02138, 800-544-4005, made many friends at the recent Apple Central Expo in Kansas City), but the programming areas on the first three are the largest and most well-known.

GEnie's Apple II areas are run by Resource Central, folks you already know from this newsletter. I run the programming area, A2Pro, on GEnie. We work hard to make sure programmers of all levels have all the tools and support they need to do whatever programming tasks they wish. We hold weekly real-time conferences at 9:30 PM ET on Monday nights where anyone can come in and chat about anything, and we hold Beginner's Night conferences at 9:30 PM ET on Thursday nights in case the other conference intimidates you. The

bulletin board is a wealth of information; as on most online services, taking the few minutes each day to read all the new messages in A2Pro will tell you more about programming than you can really imagine. The libraries are also full of the latest development information and samples as well. A2Pro also runs "A2 University," which offers free online courses to teach GEnie subscribers about programming.

America Online has a great Apple II developers area as well. Accessible by keyword ADV, the Apple II programmers' area on America Online is well-organized and has a ton of great information in it. The staff (led by Gary Jacobsen and including Jim Maricondo and Jay Fennelly) is knowledgeable and quick to help you with your problems, and there are other great Apple II areas on America Online to assist you as well. America Online's weekly development conference is held Tuesday nights at 10:00 PM ET, and the custom AOL software makes chatting really easy. ADV also has special interest groups for people who use CP/M and MS-DOS on Apple II computers, and America Online has "Programmer U," which teaches programming courses for Apple II and Macintosh computers.

CompuServe's APPPROG (Apple II Programmer) area has advantages all its own. It's not exclusively for programmers (it also houses CompuServe's Apple II sounds library, among others), and it's not as active as the other two—primarily because accessing CompuServe at 1200 bps or faster is about twice as expensive as the other online services. But CompuServe's been at this a long time. Their libraries are a wealth of information, the folks who hang out there are knowledgeable and friendly, and the staff includes Jason Harper, the author of *SuperConvert* and a whiz at graphics, C and programming the Apple IIgs. CompuServe's APPPROG area is worth your time.

Isn't all this expensive? Well, depending on how much time you spend online, it can be. Maintaining an account on all three systems costs less than $20 a month. Reading all the messages in the development forums every other day or so will take about an hour or so a week, but you couldn't get the kind of information these places offer you anywhere else at twice the price. If you can't afford to read all the online services on a regular basis, pick one and read it regularly—it's the best way in the world to get the information you need to be a good programmer.

If you have access to the Internet, or to a *Proline* BBS system near you, there's an Apple II newsgroup called comp.sys.apple2 that's often got good information. The Internet connects thousands of computers thoughout the world and moves news back and forth through a distributed system known as "news." If you're a college student, chances are good you can access the Internet (and comp.sys.apple2) for free or relatively inexpensively (when I was in college, Internet access cost $35 a semester at my school). If you're not in college or not near one, sometimes you can get access to Internet newsgroups through public access bulletin boards near you, or through a Proline BBS. Proline is an Apple II-based bulletin board system that supports Internet access. Ask at your local user group if you want to know more about connecting to the Internet.

Comp.sys.apple2 is both good and bad. It's not a programming-specific group, and since lots of students across the country have free access to it, they don't hesitate to express their opinions on subjects that barely approach the relevant. You sometimes have to wade through a lot of stuff about piracy, Email addresses, finger-pointing and more to get to the good programming information, but it is there. If you keep that in mind from the start, you'll be less disappointed by what you find there. There is no one person responsible for comp.sys.apple2 as there are leaders on the online services, so it can get very anarchical at times. If that's what you like, there's no better place.

You can also get some development information from your local Apple user group—they know about the best local bulletin boards and where local programmers can help you, and Apple sends them Technical Notes and other materials from time to time as well. Don't forget to use them as a resource if you need to find something.

Information is only the second most important thing a programmer needs. First is common sense. In my work over the years with programmers, it sometimes astonishes me how common sense takes a holiday when it's needed the most. Here are some common sense programming principles I've seen ignored very often over the years.

When you need help, ask for it. Call your local BBS or an online service and go to the programming area and say "I'm trying to do this and it's not working. Here's what I'm doing. What's wrong? Help!" Someone will almost always help you within a day or so. If you don't ask, you won't get an answer—and then you're likely to just hack together a solution that's not compatible and doesn't work for everyone. If people didn't want to ask and answer questions, programmer forums wouldn't exist on online services. Use them.

Don't use outdated stuff when newer stuff is available. The $10 you save by not updating your stuff will seem pretty trivial after you spend four days looking for a bug that's fixed in the latest book, or Technical Note, or compiler.

Pay for what you use if you expect others to pay for your stuff. Programming is not free—the work of other programmers is not yours to use freely if you think it's too expensive. If you want to write something and give it away, that's your choice—please respect the choices of those who need to make money from their work and pay for what you use.

Complicated programming isn't easy. I've seen lots of talented people give up out of frustration because they couldn't master the Apple IIgs, a system that takes about 8 books to document for programmers, in a few weeks. There are certain kinds of programming that are not easy, and the Apple IIgs has a few of them. If you're new, you will not master it overnight, no matter how quickly you picked up Applesoft BASIC or other programming languages. Don't expect to do so. And don't try to learn too much too fast, or you'll just frustrate yourself. Take your time; learn at your own pace if you need to learn.

And above all else, have fun. If you don't enjoy it, it's not worth doing. Relax, eat some hacker food and write what you want to write. If you enjoy it, you've accomplished your goal. The rest is just icing on the cake.

(Aside from being an integral part of the Apple Developer Technical Support staff, Matt Deatherage is an all-round kind of a guy, who plays the piano in his spare time.)

KansasFest is...

by Ellen Rosenberg

KansasFest is a developers conference. KansasFest is a learning (some would say religious) experience. KansasFest is a place to meet the people you've grown to know and love (or not) online. KansasFest is a gathering place for people of all ages; from all walks of life, whose commonality is a love for their Apple II's and who refuse to take no for an answer. Above all else, KansasFest is just plain fun.

This year's *A2-Central Summer Conference* was all these things and more. This year, for the first time in four years, it seemed that the atmosphere was one of acceptance rather than wild anticipation. No new rumors preceded the event, no false hopes of an impending marketing blitz or new machines that would never materialize. Rather, a time to reminisce with old friends and dignitaries; to look ahead with satisfaction at the new and anticipated products on the horizon.

As has been the tradition for the previous two years, the actual conference was preceded by two days of college courses. Participants

benefited from the knowledge of leaders in their respective fields. Mike Westerfied led the Pascal College, Walker Archer rolled in on his motorcycle (not exactly your typical biker) to teach Pascal, Bill Heineman and Nate Trost hosted a large number of attendees for the Graphics and Sound College, while Della and Jeff (Mr. SimpleScript) Smith taught the ever-popular HyperStudio College. One example of the fun-filled atmosphere was the end product of the C College which was said to have poked fun at the languages' competitor, Pascal. Hmm, I wonder how Westerfield will retaliate next year?

KansasFest itself was kicked off with Tom Weishaar's surprise interview with Steve Wozniak via the telephone wires. It was videotaped on both ends, then professionally cut and spliced together. Tom went to great lengths to obtain this interview *(how great, you may never know)* but it was well-worth the effort. Wozniak recounted Apple's beginning, reminisced, named names and told some very interesting stories. The telephone interview was a close second to what would have been his actual presence at the conference. We did invite him and from his reaction, it seemed as if he really would have come had it not been for a previously scheduled European vacation. *(Well, if I had the choice between Kansas and Europe...)*

Following Tom's act was Apple Computer's Tim Swihart with a new title. Swihart, previously the product manager of the Apple II Business Unit, is now the manager of the newly formed Apple II Continuation Engineering Group. Tim was very direct with us as he explained the Apple II's positioning within the company.

According to the compilation of figures stemming from Apple USA's K-12 Unit, there are about one million Apple IIgs computers in the marketplace today and most of them are considered still active. It is estimated that two-thirds of the one million are ROM 01 machines and that three-fourths of the one million went into the education market. It is probable that the majority of this number are used as fast IIe's, never having been upgraded beyond the initial 512k. With this information in mind, Swihart urged developers to make their own choices as far as development of new products. Apple's marketing energies and money were going to be focused on getting Mac's into the schools, according to, but not decided by, Swihart or the Apple II Continuation Group. It seems that 200,000 Mac LC II's have been sold in the last quarter (April, May and June) compared with only a few thousand Apple IIgs'. This is how Apple's marketing department justifies its efforts toward the Mac. I couldn't help but to think that the fact that these few thousand IIgs' sold themselves (even at this point in time), without any help from their corporate parents seems to have escaped the attention of Apple's astute marketing department. In the words of the dear departed John Lennon, "Imagine....."

Okay, back to reality. Many Apple II staff members were lost in the transition from the Business Unit to the Continuation Engineering Group. Evidently, some employees left prematurely, in an attempt to dodge the cutbacks. When the the newly formed group emerged, only 6 members came to surface. For this reason, Swihart has a few few vacancies to fill.

Part of the reorganization of the Apple II group included looking for new ways to do business. One way was to move certain products to outside developers. Resource Central benefited from this decision by taking over and supporting APDA's Apple II products. Not every single product survived the transition although many that didn't do have equivalents by third party developers. For instance, if you were interested in *APW/C*, you would be directed toward *ORCA/C* by The Byte Works, Inc. (4700 Irving Blvd NW Suite 207 Albuquerque, N.M. 87114 505-898-8183). Another example, not specifically mentioned by Swihart is *APW (The Apple Programmer's Workshop, itself)*. Look to *ORCA/M* instead. One more indication of this trend is the publication of *The Programmer's Reference for System 6* by The Byte Works, Inc. This book is equivalent to what would have been ToolBox #4, had Apple written it. Swihart strongly urged programmers who were developing software to run on anything beyond 5.0.2 to buy this book! He was quite emphatic as it contains important information, not available elsewhere, (even on the System 6 CD-ROM) necessary to correctly program for 5.0.4 and later. On the surface, it might seem that Apple's relegation of such products is bad news but given the present state of affairs, I personally am grateful that they are allowing the material to be published or supported by third parties. The alternative *probably* would be far less attractive.

The Ethernet Card that was described at last year's conference is nearing beta. The obvious lag in fruition was the result of a redesign of the original card. Barring any other calamities, we *could* see it by the end of the year. The card uses FriendlyNet connectors which allow connection to various types of "plumbing." These are the same connectors that the Mac's use, making it a lot easier to mix the two in labs. This flexibility does come at a cost, however; it will be more expensive than less open-minded models. The Ethernet Card will significantly speed up crowded networks (e.g. school or office multiple machine setups) but don't expect to see much difference on your networked Apple IIgs and your Mac at home. It will require an enhanced Apple IIe or an Apple IIgs with System Software 6.0.1 installed.

The Ethernet Card will be the number one priority for features that go into System 6.0.1, which will be released concurrently. The release will be user-oriented as opposed to developer-oriented. System 6.0.1 will contain new drivers for Ethernet, updates for some of the Control Panels and bug fixes for some of the compatibility problems that were discovered after 6.0 was released. One of the projected updates that I personally find most exciting is keyboard navigation in the Finder. (Thanks to Andy Nicholas and Dave Lyons.) There will probably also be a read-only MS-DOS FST in 6.0.1 if, and only if, it is fully tested and read to go in time for the Ethernet release. Future plans include a full read/write MS-DOS FST.

More Apple bits. If a prize were given to the session with the most attendees standing for lack of seats, the award would go to Bill Heineman's Avatar session. (It was a no-win situation for the three other session leaders who were up against this one!) It will be real interesting to see how this idea develops. Simply put, Avatar is intended to be an Apple IIgs compatible system built around mostly non-proprietary hardware to keep it inexpensive and expandable. The hardware contraints may limit its ability to be absolutely 100% compatible with some Apple-proprietary standards such as the Apple Desktop Bus or Appletalk.

Some of the most interesting non-technical sessions were those dubbed "Old-Timers." Silas Warner, author of the classic *Castle Wolfenstein* and founder of Muse Software, entertained us with interesting stories of the past for an hour. One of his most amusing stories had to do with the assembler that they wrote that made it next to impossible to disassemble the code. In another session, Roger Wagner and Alan Bird talked mostly about Bert Kersey and his slightly off-center sense of humor. The Apple IIgs sage (Roger) also recalled how he turned down the rights to publish *Print Shop* because he didn't think that anyone would buy it. Right!

Other non-technical happenings included the Thursday night roast of Roger Wagner and the Friday lunchtime skit a la Monty Python, performed by Steve Disbrow of *GS+ Magazine* and friends. The roast was truly hilarious. I can't help but to wonder if, uh, computer nerds (I say that affectionately) on other platforms are as funny as these guys were. The participants of the roast were Steve Disbrow (host), Jay Jennings (Softdisk), Matt Deatherage (Apple, Inc.), Paul Statt (InCider), Tim Swihart (Apple, Inc.) and Tom Weishaar (ResourceCentral). And of course, Roger Wagner who was so quick on the comebacks that he must have had an advanced (Simple)script. At the request of many who were there, we decided to put together a

video of the Wozniak interview including the roast and the skit as well. Aside from the entertainment value, it would make a great user group presentation. See this month's catalog for details.

There's so much more to tell and so little room to tell it in. More next month. If you want it.

Miscellanea

For a long time the Resource Central subscription system was maintained on Apple II computers. Recently this era ended. Finding subscribers and their orders was taking too long and what we needed was a finely tuned order system.

Tom Weishaar spent well over a year outlining his desired features, locating database programs, and trying to design a new system that could handle both products and multiple publications. Ultimately, the project turned out to be overwhelming and he again went hunting for a ready-made application. What Tom found was a commercial multi-user subscription system called *Multi-Pub* from a company named Datasystem Solutions, Inc., which just happened to be located in nearby Fairway, KS.

When you have exacting needs in software you end up picking the program that fits those needs first and the system it runs on second. In this case, *Multi-Pub* runs under the Unix operating system usually associated with mini and mainframe computers. We looked at A/UX on the Macintosh, balked, and instead settled on a 33MHz 80486 behemoth (affectionately referred to as "Igor") with a 16-port serial expansion interface, tape backup, SCSI hard disk with 4.5 megabyte caching controller (an expensive combination costing about as much as the CPU) and various other peripherals. We spent a couple of days assembling the pieces, a couple of weeks debugging the basic system, and are now in the process of getting acquainted with (and working the kinks out of) our *Multi-Pub* installation which was fired up the first of July. Eventually, everyone will have a terminal on their desk to allow them to look up information in the system. (Actually, the terminals are there already, it's learning to use them that's taking a little time.)

There have been a few glitches, such as duplication of subscriptions for some who also subscribed to *Nibble* (if you are getting duplicate issues, please send us the customer codes from both subscriptions and we'll get it fixed) and some munged addresses (if your address information is not correct, let us know).

All in all, things should get better once we get over the rather substantial hump of learning (a) a new hardware platform, (b) a new operating system, and (c) a new subscription system, all within the space of a few weeks (the addition of *Nibble* subscribers forced this all into play a bit sooner than we had anticipated). The most important contributions of the new system are to free Tom up to spend more time overseeing his company's publications, both old and new, and speeding up answers to inquiries about orders.—DJD

We finally got an Apple 3.5 Disk Controller Card "in house". This $149 (sugested retail) interface allows you to connect any of Apple's 3.5 drives for the Apple II (UniDisk 3.5, Apple 3.5, or high density "SuperDrive") to the Apple IIe or IIgs computers.

The package includes the interface card with installation hardware, a 53-page user manual, and a disk with an "updated" GS/OS APPLEDISK3.5 driver (for IIgs System Software 5.0.4; System 6.0 already includes the updated driver) for recognition of the SuperDrive formats and a set of IIe-compatible utilities.

The card itself is about a "half length" board with a red activity indicator light along the top edge. The card includes 32K RAM to hold buffered data; subsequent reads require starting the drive and accessing the physical disk only if the requested disk information isn't already in the card's buffer. This should both increase speed of disk access and lower the drain on the computer power supply (starting the drive motor and spinning it up to speed requires some effort on the part of the power supply).

The manual contains detailed installation information with a couple of glitches. It says the card cannot be used with a UniDisk 3.5 on the IIgs; we were told at the ***A2-Central Summer Conference*** that it could now be used with all three types of drives (you will need to install the current drivers from System 6.0 for the UniDisk 3.5 if one is attached). Also, the manual declares that you can't use the controller in slot 7 if you use AppleTalk on a IIgs; this is only true for a ROM 01 machine (on a ROM 03, AppleTalk can be assigned exclusively to slot 1 or slot 2).

The card can be installed in any available slot; most commonly you'd use slot 5 since that's where a 3.5 controller is usually expected to appear (but programs should allow for presence of the controller in any slot). On a IIgs you install the card in slot 5, change the setting for slot 5 to "Your Card" (instead of "SmartPort"), then connect your 3.5 drives to the new controller. Older 5.25 drives remain connected to their controller (usually in slot 6); on the IIgs you would disconnect any daisy-chained 5.25 drives from the last 3.5 drive and connect them to the IIgs disk port on the back of the computer, and leave the setting for slot 6 in the IIgs control panel assigned to "Disk Port".

From there, the card works just like your current disk interface. The exceptions you run into are that, if you have a SuperDrive, you can now format, read, and write high-density 3.5 disks in the 1.44 megabyte format. These disks have an extra hole punched through the upper left-hand corner (opposite the normal 3.5 write-protect hole) to indicate they are high-density disks. You can install ProDOS or the IIgs System Software on high-density disks to serve as boot disks.

You can't read high-density disks on the Apple 3.5 or UniDisk 3.5, however, so consider this when making disks you intend to be used on other configurations. Also, you may see kits to "convert" double-density disks to high-density by punching the second hole to "fool" high-density drives; this is not recommended since the recording media properties do differ between the double- and high-density disks. (What you write on the disk may stick around a short while but may not be there a couple of years down the road.)

There are third-party drives that are mostly compatible with the SuperDrive. More recent versions of the AMR 3.5 drive we've seen use such mechanisms, and they can be used on the Apple 3.5 Controller Card. However, on the IIgs they don't respond exactly like the Apple drives when the on-line devices are "polled" for the presence of volumes and the computer will appear to "hang" if there isn't a disk in the drive. We've learned to live with this by keeping a blank disk in the drive; when a disk is present (or inserted) the "hang" clears and the computer can continue.

The SuperDrive and Apple 3.5 Controller combination opens the way toward reading and writing IBM PC compatible 3.5 formats. The PC uses a different recording method than Apple's standard drives which the SuperDrive also supports. The SuperDrive and controller combination is intelligent enough to make the format transparent to the computer; you can issue a command to read a physical block of data from the disk and it will be delivered regardless of the disk format (ProDOS, Mac, or MS-DOS format; 800 kilobyte, 720 kilobyte, or 1.44 megabyte densities). The task of writing a utility that can access MS-DOS disks is reduced to writing routines that can access the MS-DOS file system structure. We've already seen a ProDOS 8 utility that can read files from the root directory of an MS-DOS disk using this technique, and Apple engineers mentioned at the ***A2-Central Summer Conference*** that the feasibility of an MS-DOS File System Translator for GS/OS was being investigated.—DJD

This isn't exactly new news and it's certainly not good

news, but I just recently read it in a few user group newsletters. It seems that the plans for the commercial version of *Soundsmith* have come to a halt. Evidently, at least one beta copy that was being tested by members of the Seven Hills Partner Program got in to the wrong hands. The program was then uploaded to numerous pirate boards around the country. and perhaps the world. The result was the loss of one more Apple IIgs program for the rest of us. No additional comments or sermons necessary here.

One program I've seen recently that does belong on every Apple II board in the world is a freeware stack based in HyperCard IIgs by Cynthia Field. It's entitled, *Better Safe than Sorry,* and it stars McGruff, the crime fighting dog. Those of you with young ones ought to recognize this character, who is a symbol for crime and drug prevention among school children. Not only is the stack interactive and informative but it also contains a fun activity at the end of each segment. It's sure to hold the interest of young and old alike. By the time you read this it should be on the major online services and perhaps in user group libraries. If you have trouble finding it, you can send $10.00 (for shipping and handling) to The Rhode Island Crime Prevention Officers Association, 40 Caswell Street, Narrangansett, R.I. 02882.

Spear-headed by GEnie's Tim Tobin, a search has begun to locate the authors of discontinued Apple II software with the intent of entering the programs (and the related source code, if possible) into the public domain, or getting them declared as either shareware or freeware. The project, known as Lost Classics, began some months ago but kicked into high gear in late July to coincide with KansasFest. It already has had at least one major triumph. Paul Lutus, the elusive author of the renowned *Apple Writer 2.1* word processing program, was found and convinced by Tobin to declare his program freeware. If you can contribute to this project or know someone who can, contact Tobin on GEnie (A2.Tim) or phone him at 301-813-5697.

***The Programmer's Reference for System 6.0* by The Byte Works, Inc. has been referred to several times in this issue.** It covers all of the changes, enhancements and additions to the Apple IIgs operating system since System 5. Included in the 478 page manual are new tool calls and tool updates; documentation for the Midi Synth, Media Control Tool Set and the Video Overlay Tool Set; Finder documentation; GS/OS updates; information about the new FSTs; Sound Control Panel documentation; and a complete concordance listing every page a tool is documented over all four volumes of the toolbox reference manuals.

It may come as a surprise that Roger Wagner Publishing is in the process of developing a Mac LC version of HyperStudio. According to Wagner, the project is well on its way with anticipated delivery time the spring of 1993. Although, feature for feature, it will be very similar to the Apple IIgs version, he still believes that the IIgs is the best multimedia machine around, both in cost and performance. Designed specifically for the Mac LC, it will run on other Macs as well. It will come with a retail price tag of $229; lab packs, network, and site license versions also available. For further information contact RWP, Inc., 1050 Pioneer Way, Suite P, El Cajon, CA 92020.—edr

Ask (or tell) Uncle DOS

Forewarned

In recent issues, both *A+/inCider* and ***A2-Central*** printed subscriber letters which state the HFS FST works fine under System Software 5.0.4.

This is dangerously incorrect.

While Apple tries to keep the internals of GS/OS fairly stable from release to release to minimize the opportunity for errors, GS/OS has always required substantial internal changes when new read/write file systems are added. This was true in System Software 5.0 for AppleShare and is true in version 6.0 for HFS.

The HFS FST may appear to work in normal circumstances (as the letters indicate), but the older versions of GS/OS do not meet the demands of the newer FST. The FST will almost certainly crash if you try to initialize any disks.

Components from different system software versions do not mix and match — do not use older system software files with newer system software versions. GS/OS drivers are an exception — they have a stable, documented interface and will work fine with any version of GS/OS later than the one for which they were designed, although some manufacturers may require newer features of newer drivers. (For example, the System Software 6.0 Apple 3.5" driver supports SuperDrives attached to an Apple II 3.5 Disk Controller Card — while you can use the older version of the Apple 3.5 driver safely with System Software 6.0, you won't be able to use drives connected to an Apple 3.5 Disk Controller Card.)

Mixing and matching system software in general is dangerous — but using the HFS FST under 5.0.4 will eventually crash and it may trash your disks. Do not attempt this with any disks online that you want to keep.

Matt Deatherage
Apple Computer, Inc.

System 6 counseling

I have a couple of pieces of software that don't work with System 6. One is a hard drive installed version of *Battle Chess* and the other is a rare 8-bit program that I still like to use. It's called Appleworks.

Seriously, the problem with Appleworks is that ProDOS 2.0.1 and older versions of the RamFAST SCSI card are incompatible. Drew Vogan of CV Technologies apologized that that he never checked compatibility of the older ROM's with System 6.0 and promised to send me a free upgrade to 2.01c chips. He is also sending me details of the $15.00 upgrade for the new 3.0 chips. I'll probably buy them, sight unseen because CV Tech has been one of the Apple II's great supporters and I want to support them.

Why does the desktop vanish when going from 5.0 to 6.0?

Barry Austern
Cincinatti, Ohio

Bill Heineman is working on a GS/OS launchable version of Battle Chess. Watch this space for updated information.

The AppleWorks desktop doesn't disappear, it sometimes just argues with ProDOS about disk locations. AppleWorks tries to locate all disks when it starts up and that results in some arbitration.

One problem relates to the ProDOS 8 v2.0.1 device remapping; now that ProDOS 8 remaps devices, it's best not to have another entity also trying to shuffle them around. If you're using a RamFAST, use the RamFAST utility program to assign all SCSI disk volumes except those for the slot the RamFAST is installed in to "slot 0" (ROM 2.0) or "Unmap volume" (ROM 3.0). That leaves the remapping to ProDOS

Here's another wrinkle: A problem with the existence of over 14 ProDOS 8 disk devices had been reported. Apple engineers discovered that the problem isn't a bug in the new ProDOS 8. At KansasFest Matt Deatherage announced that they had found an error in the Apple II Technical Note's code example for unhooking and reconnecting the /RAM volume. Programs which use this code exhibit problems; the code will be corrected in an upcoming technical note and hopefully existing programs can be patched to fix the glitch.

Counting the 14-device "limit" for ProDOS 8 is not always straightforward. Disk II-style 5.25 controllers (including the slot 6 "Disk Port" setting for the IIgs, if enabled in the Control

Panel) will always report two 5.25 drives being present even if fewer are connected because the Disk II drives don't report a status to indicate they exist.

Next, AppleWorks can't look at 14 devices at once anyway. It's limited to 8 devices, so the "bottom 6" will be shoved off the guest list. If you're missing a volume you think should be visible, that may be what happened.

Finally, AppleWorks doesn't like to recognize printers in slots where disk devices are assigned. If you try to print to a slot where you know you've installed your printer interface and AppleWorks balks, check to see if a disk volume is assigned there in AppleWork's list of on-line volumes. If so, you'll need an AppleWorks patch to make AppleWorks ignore the presence of the phantomed volume (it's safe to print as long as there is a printer interface in the slot).This patch can be found in ***SuperPatch*** *v. 6.1 by John Link (Quality Computers, 1-800-443-6697).—DJD*

The installation of System 6.0 on my Apple IIgs went as expected. However when I attempted to load Appleworks 3.0 to the desktop, it would either lockup or go into the monitor.

Your office suggested that the problem was probably related to my verson of the CV Tech RamFAST SCSI card. I tried just about everything. but there were no ProDOS 8 programs that would run from System 6.

For some reason, I have been unable to get a response from CV Tech. Finally, after several letters, Tom Weishaar came up with a possible solution.

Go into GS/OS/System/P8. Find Block 5, Byte $1A3; change $A5 to $00. (The problem is related to ProDOS 8 rewrapping devices—the RamFAst does this as well.)

I was able to get into P8 by using *Salvation-Deliverance*. Pull down the File menu and use the Editor. Access /GS/OS/System 8.

However, the block to edit when using *Salvation-Deliverance* is not Block 5, it is Block 4. Follow the directions on the program, you will get a "Reboot" message if you have made the change correctly.

Ephraim Wall
Perkins, Okla

It's not news that earlier versions of the CVTech SCSCI card are incompatible with the latest system software. They are a small company and were probably deluged with phone calls and mail once System 6.0 hit the streets. Therefore, the spotty response. Thanks for sharing your solution.—edr

Thanks to Dean Esmay, last month's ***A2-Central on Disk*** includes a fascinating extract from GEnie highlighting various problems some people have encountered installing System 6.0.

John Mayotte's experience was especially edifying in that I have exactly this problem; System 6.0 not recognizing a 20SC hard drive with ROM 01 and Apple Rev C SCSI card. Although I had also discovered that copying an earlier version of the SCSI.Mangager and SCSIHD.Driver (in my case from 5.03) to my 3.5 System 6.0 allowed me to acces by hard drive, I'm wary of using this non-standard set-up; and certainly unwilling to install on my hard drive in case there is some underlying incompatibility between these drivers and the new system. I have found, for example, that when I re-boot from my hard drive under System 5, the Finder information is lost.

Are my fears unfounded, or would you recommend upgrading to either the Apple Hi Speed SCSI card or the RamFAST SCSI card? What are their various merits?

Any comments would be very welcome. Dean has included a number of very interesting items on the last two disks that I cannot use because they require System 6.0.

Bill Seabrook
Tyne & Wear, U.K.

We ran into the same problem; fortunately the user was local so we could look at the hardware. The problem seems to be the response of the SCSI drive itself to an inquiry from the System 6.0 SCSI drivers, not the interface. If you want to upgrade your SCSI interface that's fine by us but we just don't know if changing interfaces will help.

We made copy of the System 6.0 "/Install" disk and then replacing the 6.0 "SCSIHD.Driver" and "SCSI.Manager" files (in the "System/Drivers" folder) with the "SCSIHD.Driver" and "SCSI.Manager" files from the 5.0.4 "/System.Tools" disk. We used this disk to boot and it recognized the hard disk and allowed installation.

This is not an Apple-tested configuration so they probably can't guarantee this setup, but it seems to be working in this instance. The Installer scripts all seem to copy the SCSI hard disk setup files from the "/Installer" disk so any further installations should be consistent. The 5.0.4 SCSI drivers also lack some of the features of the 6.0 drivers (like being able to write-protect a volume in it's "Icon Info" dialog).

Disk drivers are not FST's, but it is still not recommended to mix and match System Software versions (see Matt Deatherage's comments earlier). We suggest it in this instance only because it seems to work (and the problem has been reported to Apple so that they can look into it).—DJD

Transfer tickets

In the March and April 1991 issues of ***A2-Central*** (pp 7.16b-c and 7.23c-241) there were letters relating to using Tandy's WP-2 portable computer as a portable Apple II. On the basis of those letters and articles by Dr. David Thornburg in *InCider/A+*, I began searching for a portable computer.

I compared the Tandy and the Laser PC-4 (an upgrade to the PC-3) computers. Both use similar processors (280). While the Tandy had a larger screen and a built-in thesaurus, the Laser computer was cheaper by about $100.00, has 2 megs of ROM, and the RAM could be expanded to 128k vs the Tandy's 64k expansion capability. I bought the Laser PC-4.

My primary use for the PC-4 is writing letters while travelling or camping. It also serves as an electronic notepad for taking minutes of committee meetings. My third grade son loves to use it to compose adventure stories (precisely what Dr. Thornburg envisioned). Once I get home, I connect the RS-232C serial port on the PC-4 and the modem port on the Apple IIgs using a combination of Laser's cable and the *Crossworks* cable. I use *Snowterm IIgs* modem software ($20.00 shareware) and the built in serial upload capabilities of the PC-4 to transfer text files to the IIgs. Even at 19200 bps, it works wonderfully well.

I've had two problems. The first is speed. The 280 processor is rated at 3.58 mhz but except for launching programs is much slower than the 1 mhz Apple IIe—I can live with that for the price. The other problem that I can't seem to transfer files from the IIgs to the PC-4. I've tried several different communications programs at various settings and have not had any luck. The PC-4 always gives me a message saying that an error has occurred and refuses to accept the file. Perhaps someone with either a Tandy WP-2 or a Laser PC-3 or 4 could help me with this.

Bob Allen
Limuru, Kenya

A2-Central™

[redacted]

Publisher: **Tom Weishaar**
Editor: **Ellen Rosenberg**

with help from:
Denise Cameron **Dennis Doms** **Sally Dwyer**
Dean Esmay **Richard Ginter** **Jeff Neuer**
Jean Welshaar

A2-Central,—titled ***Open-Apple*** through January, 1989—has been published monthly since January 1985. World-wide prices (in U.S. dollars; airmail delivery included at no additional charge): $34 for 1 year; $60 for 2 years; $84 for 3 years. All back issues are currently available for $2 each; bound, indexed editions of our first six volumes are $14.95 each. Volumes end with the January issue; an index for the prior volume is included with the February issue.

The full text of each issue of ***A2-Central*** is available on 3.5 disks, along with a selection of the best new public domain and shareware files and programs, for $90 a year (newsletter and disk combined). Single disks are $10.

Please send all correspondence to:

A2-Central
P.O. Box 11250
Overland Park, Kansas 66207 U.S.A.

ISSN 0885-4017 **GEnie mail: A2-CENTRAL**